ACEEE Int. J. on Recent Trends in Engineering and Technology, Vol. 10, No. 2, Jan 2014

Implementation of BDDs by Various Techniques in Low Power VLSI Design

Purnima Kumari Sharma[1] and Manish Kumar[2]
North Eastern Regional Institute of Science & Technology
Department of Electronics and Communication Engineering, Nirjuli, Itanagar, Arunachal Pradesh, India
purnimasoni4018@gmail.com, er.manish.k@gmail.com

***Abstract*— Power has become an important design parameter in today's ultra low sub-micron digital designs as found. The impact of the increase in power is multi-discipline to researchers ranging from power supply design, power converters, voltage regulators design, system, board and package temperature analysis and signal integrity analysis to minimizing power itself. This survey basically focuses on steps taken till date in solving the challenges arising due to increase in power in CMOS digital circuit design using a very efficient pre-computation technique with the use of BDDs (Binary Decision Diagrams).**

***Index Terms*— BDD, MUX, CMOS, Dual rail, PTL, Adiabatic, Kernel**

I. INTRODUCTION

In the last few decades there have been a very vigorous research looking from the power efficiency point of view. Increasing technology and reduction in semiconductor sizes of MOS devices have forced people to find out efficient methods for power reduction. Many researchers have taken initiative to work in this field and got efficient results. Still the methods have not reached the zenith of excellence to stop working in this field. Many of them used BDDs as an efficient tool for power reduction .A Binary Decision Diagram (BDD) is a directed acyclic graph (DAG) that represents a Boolean function (or multiple functions) as a sum-of-disjoint product (sodp) form [12].

Mathematical foundation for BDDs is the Shannon decomposition theorem:

$$f(x_1,x_2, ...,x_n) = x'_1 \cdot f|_{x1=0} + x1 \cdot f|_{x1=1} \quad (1)$$

where

$f|_{x1=0} = f(0,x_2, ...,x_n)$, $f|_{x1=1} = f(1,x_2, ...,x_n)$ are subfunctions of f, called co-factors.

By using (1), a BDD is formed like a tree (also called binary decision tree) for any Boolean expression. Transformation of a binary decision tree into a BDD is illustrated in Fig. 1. In this figure, dag (b) is obtained from dag (a) by merging isomorphic subdags rooted at 1. Dag (c) is obtained from dag (b) by removing a redundant test on r. Dag (d) is obtained from dag (c) by merging isomorphic subdags rooted at q. Dag (e) is obtained from dag (d) by removing a redundant test on p. Finally, dag (f) is obtained from dag (e) by merging isomorphic subdags rooted at 1.

A BDD is called ordered (OBDD) if each variable appears at most once on each path from the root node to a terminal node, and if the variables appear in the same order in all other paths. An ordered BDD is called reduced (ROBDD) if it is devoid of any isomorphic subgraph or any redundant node. As the ROBDD representation is unique for a Boolean function with a given variable ordering, it is considered as canonical representation of the function for a particular variable ordering.

It is seen that not only BDDs provide an efficient data structure to represent Boolean functions, there is one-

DOI: 01.IJRTET.10.2.525

to-one correspondence between a BDD and a MUX-based realization of the function. So, the number of MUXs required to realize a function depends of the number of nodes in the BDD, which in turn depends on the ordering of variables. This helps in reducing power to a greater extent.

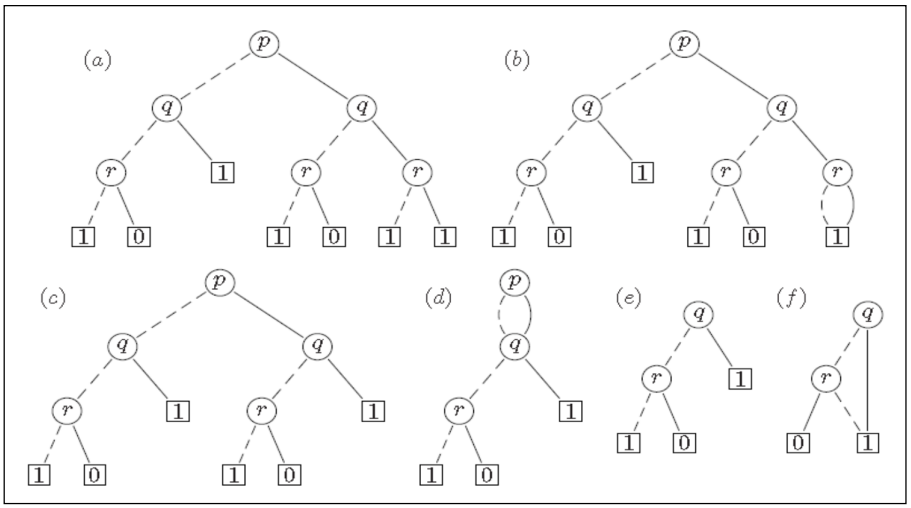

Figure 1. Transformation of a binary decision tree into a BDD

II. Power Reduction Techniques Using BDDs

A.BDDs for mux based FPGA

This approach for power reduction in FPGAs comprises two basic steps as shown in Fig. 2. The two basic steps are optimizing decomposed BDDs with the help of ratio parameter based heuristic and then technology mapping of the optimized BDDs onto FPGA cells [1]. This RP heuristic tends to produce balanced BDDs, which results in an increase in the number of leaf nodes, giving rise to greater number of signal paths from root to leaf. Thus the area increase does not result in increased fan-out in the signal path. Techniques like node duplication and sharing have been applied to minimize the number of FPGA cells and delay during technology mapping. The cell configurations have been chosen such that the switched capacitance and hence the power dissipation is minimized.

In [1], the proposed algorithm was tested on a large number of several ISCAS benchmark circuits. The result, in terms of area, represented by the number of FPGA cells is found to be comparable, but the performance in terms of delay and energy (power-delay product) are superior to the existing reported results.

BDDs generated by this approach do not have the same ordering of variables at the same level along different paths. These BDDs may be termed as Reduced Unordered BDDs (RUBDDs), in contrast to Reduced Ordered BDDs (ROBDDs) commonly used in the existing approaches.

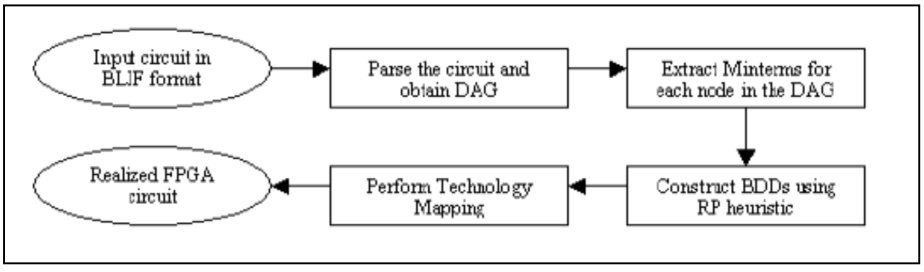

Figure 2. Basic Steps for Power Efficient mux Based FPGAs

B. BDD-based Logic Synthesis Using PFAL Adiabatic Multiplexer

PFAL multiplexer as shown in Fig. 3 is found to give better results than any other adiabatic circuits [8]. In contrast to the existing approach of mapping each node of a ROBDD to one 2-to-I MUX block, the inherent dual-rail feature of the adiabatic MUX circuits has been exploited to reduce the number of MUX blocks dual-

rail feature of the adiabatic MUX circuits has been exploited to reduce the number of MUX blocks required in the implementation and increasingly larger reduction for lower frequency of operation.

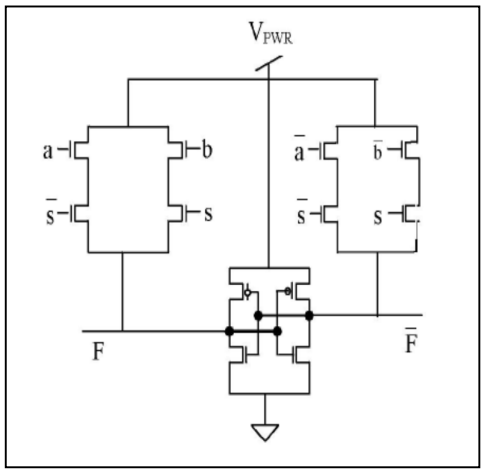

Figure 3. PFAL Dual Rail Adiabatic Mux

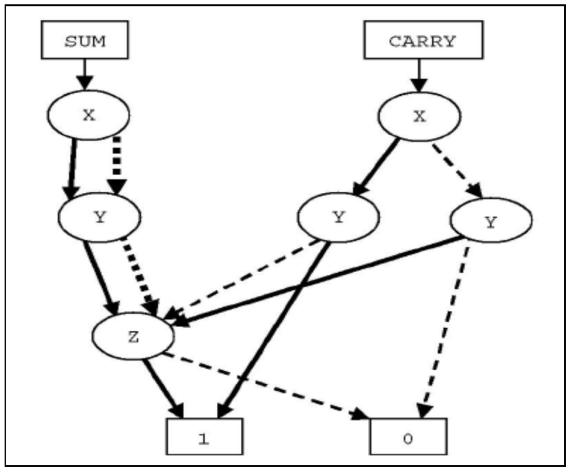

Figure 4. BDD drawn by collapsing complementary pairs

Fig. 4 illustrates reforming of the original BDD by collapsing complementary pairs. Each of these pairs can be replaced by a single adiabatic MUX block for technology mapping. Each of the remaining nodes will also need an adiabatic MUX, though not fully utilized. The final full adder circuit can be synthesized using adiabatic MUX blocks as shown in Fig. 5. Result shows, on an average, 22% reduction in number of MUX blocks and over 50% power consumption in the reduced BDD based adiabatic circuit [2].

C. BDD-based Synthesis Using Dual Rail Static DCVSPG Logic

A proposed algorithm [7] for low power digital circuit design is applied on MCNC benchmark circuits and observed considerable reduction in the number of BDD nodes after collapsing complementary pairs. It is seen that the modified BDD is then directly mapped to DCVSPG-based MUX cells as shown in Fig. 6. Each of these MUX cells was found to be simulated using CADENCE SPECTRE SPICE on 180 nm technology with the supply voltage of 1.5V and 500 random input patterns and the results showed an average of 54% savings in energy (in fJ) for frequency ranging from 1 MHz to 1 GHz

Moreover, by experimental analysis various parameters were calculated on an average and found 22% reduction in number of MUX cells and 65% reduction in power consumption in the modified circuit if compared to the circuit synthesized by static CMOS MUX cell. As the number of transistors is same (12 transistors) for both static CMOS MUX cell and DCVSPG-based MUX cell, the average reduction in number of transistors for all benchmark circuits was also taken as 22%.

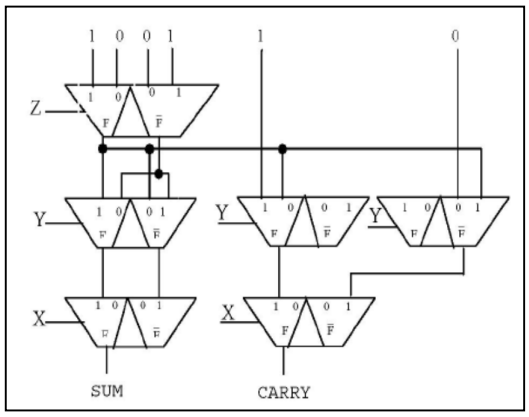

Figure 5. Full adder synthesis using adiabatic MUX

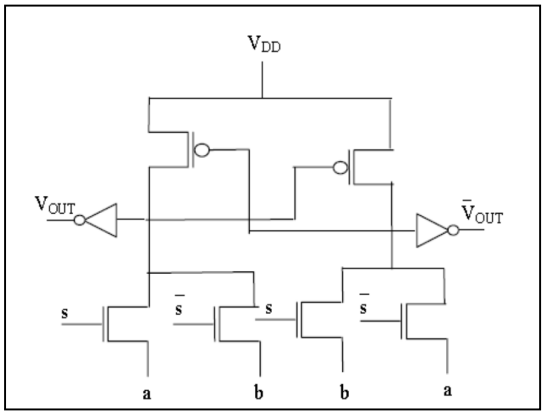

Figure 6. Dual Rail DCVSPG Mux Cell

D. BDD-based Synthesis Using PTL logic

In this method, the BDD of any Boolean expression is mapped to a pass transistor circuit with a straightforward transformation. Fig. 8 shows mapping of BDD nodes shown in fig. 7 to pass transistor multiplexers. Only nMOS transistors in the pass transistor network are used. Advantages of this choice are that the input load capacitance is minimized (faster circuits and lower power dissipation), and that the area is reduced as well. However, there is need of both input polarities to drive the pass transistor multiplexer.

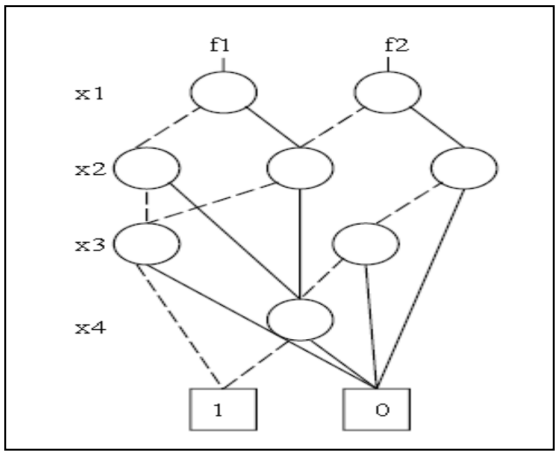

Figure 7. Multi-rooted BDD

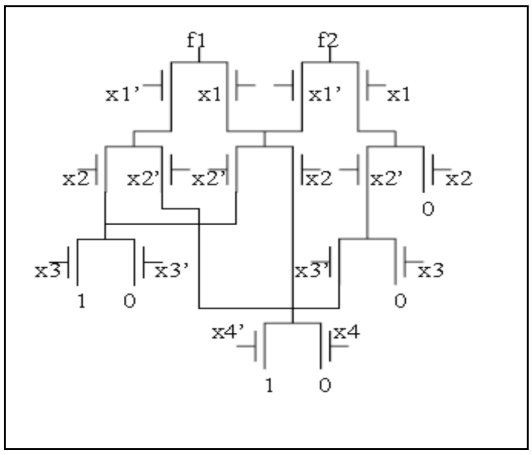

Figure 8. Pass transistor mapping

It is found that the node count is a good measurement for the complexity of the BDD: for a node count up to at least 500 the direct mapping of BDDs to layout seems feasible [9]. When using double-sized pass transistors it was found that the results were sometimes better, but not always. This indicated that the optimum size for the pass transistors is not always the minimal size, but should be chosen according to the performance, power and area trade-off. An efficient power reduction is seen in this case. Synthesized designs obtained using BDDMAP [3] on a subset of MCNC91 logic synthesis benchmarks when compared generally had better area-delay and power-delay products compared with synthesized designs for the same benchmarks generated with Design Analyzer, a commercial logic synthesis tool from Synopsys.

E. Kernel based circuit synthesis using BDD

The proposed algorithm [4] for low power circuit synthesis consists of three steps: BDD extraction, kernel selection, and a circuit synthesis process. The proposed circuit structure is based on the transformation of a given function f using an extracted kernel as follows. A function f can be expressed with respect to a kernel K as $f = f_k . K + f_{k'} . K'$, where f_k and $f_{k'}$ are generated by dividing the given function f by kernel K and K´ respectively.

The Kernel selection step chooses the best partitioning solution in terms of the cost function out of possible partitioning solutions using a simulated annealing algorithm. The cost function has been derived to select the best kernel that can lead to reduction of power under area constraints. The kernel search is performed using BDD, and its cost function is calculated by the estimated circuit power using the switching activities of roughly generated sub circuits or their BDD sizes. The kernel based pre-computation structure is shown in Fig. 9.

While doing experiments using the proposed algorithm it was found that, if the signal probability of the input variable was taken to be 0.5, the power dissipation of the circuit in Fig. 10 was 45 µW. The circuit obtained by the proposed scheme as shown in Fig. 11 consumed 28.2 µW, showing improved performance. It was seen that the literal count of the circuit by the proposed scheme was 12, compared to 16 in the circuit generated by SIS. The example shows that the proposed algorithm reduces area. The reduction of power dissipation in the circuits generated by the proposed algorithm was seen to be 27.9% less than the original circuits optimized for area and 22.5% less than the circuits based on the pre-computation scheme.

F. NAND BDDs

A BDD package is presented that is tuned for fast and efficient combinational equivalent checking. Instead of using the three-operand ITE operation, the basic synthesis algorithm used here is the two-operand Boolean

NAND. This operation simplifies the implementation of the software and also has advantages regarding the hit rate of the computed Table I and also reduces its size.
The realization of the package described here was based on the principle that all operations that are not relevant for the computation of the BDD are avoided. Moreover, it is not a "full" BDD package since few features like dynamic variable ordering and memory management are missing. It was found by testing using benchmarks from ISCAS85 that the number of nodes is never more than 50% larger for NAND-BDDs as compared to the ITE based implementation while the runtime in some cases can be reduced by more than 99% [10].

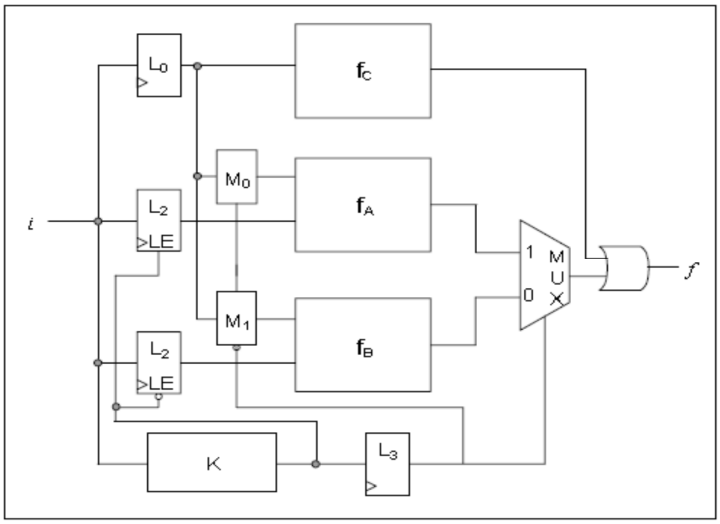

Figure 9. The kernel based proposed pre-computation structure

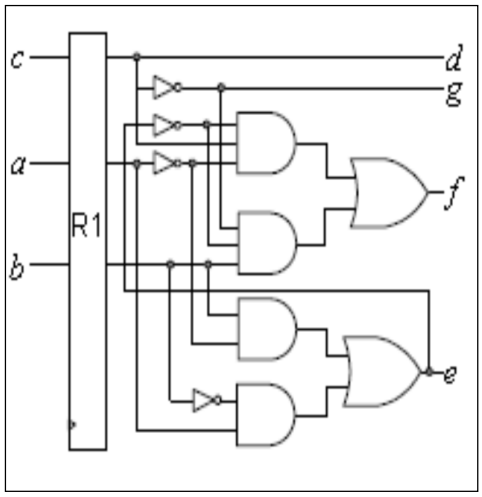

Figure 10. An example benchmark circuit 'b1'

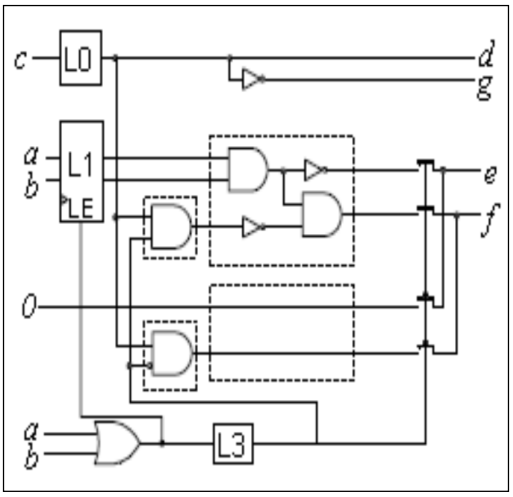

Figure 11. Circuit obtained by the kernel based algorithm

G. NOR BDDS

An approach for the realization of a BDD package, which uses two operand NOR operation instead of using three operand ITE operation to perform manipulation of Boolean functions is designed. This method does not consider some of the main features of BDD package such as dynamic variable ordering, complemented edges, etc. The modified table for NOR operation is given in Table II.

This method works with 2 operand compare to ITE operation, so, the execution of the BDD operations is faster. No complemented edges are considered in this method. So the memory requirement for this method is

lesser than other operations and moreover glitch power is minimized [11].

TABLE I. REALIZATION OF OPERATORS BY NAND OPERATION

Function	Name	Expression	NAND-call
0000	0	0	0
0001	f AND g	f . g	NAND(NAND(f,g),1)
0010	f > g	f . g´	NAND(NAND(f,g´),1)
0011	F	f	F
0100	f < g	f´ . g	NAND(NAND(f´,g),1)
0101	G	g	G
0110	f XOR g	f ⊕ g	NAND(NAND(f,g´), NAND(f´,g))
0111	f OR g	f + g	NAND(f´,g´)
1000	f NOR g	(f + g)´	NAND(NAND(f´,g´),1)
1001	f XNOR g	(f ⊕ g)´	NAND(NAND(NAND(f,g´),NAND(f´,g)),1)
1010	NOT f	g´	g´
1011	f ≥ g	f + g´	NAND(f´,g)
1100	NOT g	f´	f´
1101	f ≤ g	f´ + g	NAND(f,g´)
1110	f NAND g	(f . g)´	NAND(f,g)
1111	1	1	1

This method does not provide a fully fledge BDD package mainly because of the following reasons:

- Combinational Equivalent checking needs very fast execution of larger designs. Variable ordering might result in an increase of the time complexity. A heuristic Ordering might solve this problem.
- Complemented edges are not considered, therefore no comparison between the variables are required. It leads to minimum use of memory for the BDD package.

Advantages of using this system can be classified as follows:

- Easy to implement compared to any other logic function mainly due to the fact that NOR is a universal gate.
- This method works with 2 operand compare to ITE operation. So the execution of the BDD operations will be faster.
- No complemented edges are considered. So the memory requirement for this method will be less than other operations.
- This will fit for combinational Equivalence checking of larger circuits with less time complexity, since numbers of nodes are not counted.

Variable ordering techniques are not required.

H. BDD based design using hybrid Domino Pass Transistor Logic CMOS

Asynchronous system design represents an important design methodology in the recent deep sub-micron technologies. The high speed and low power solutions are possible with asynchronous design techniques using simple handshaking and completion detection logic.

In [6], the design of a hybrid Domino Pass Transistor Logic CMOS (PTL-CMOS) based 2-bit asynchronous adder is presented. The PTL part of the whole design is designed using the BDD principles. Also, using the former design as basic building block of an 8-bit asynchronous adder has been implemented. The simulation results showed a reduction in number of transistors over Minimal Energy Dual-bit Dynamic adder (MEDB) adder without any compromise in the delay.

The hybrid BDD-based design for an 8-bit asynchronous adder was simulated using DSCH tool and also cadence tool, UMC 180nm, 1.5V technology. It showed a total of 268 transistors whereas this number was 376 in the case MEDB adder while considering the number of transistors in the logic block only. Thus, there was a great reduction in the number of transistors in BDD–based design. The pre-charge signal was driven through an inverter chain so that it can drive large loads. A very important point to note in the asynchronous BDD adder was that there was no static short circuit path due to the use of BDD, while such static short-circuit path can be found in DBPTL logic based design.

TABLE II. REALIZATION OF OPERATORS BY NOR OPERATION

Function	Name	Expression	NOR-call
0000	0	0	0
0001	f AND g	f . g	NOR(f´,g´)
0010	f > g	f . g´	NOR(f´,g)
0011	F	f	f
0100	f < g	f´ . g	NOR(f,g´)
0101	G	g	g
0110	f XOR g	f ⊕ g	NOR(NOR(NOR(f,g´), NAND(f´,g)),1)
0111	f OR g	f + g	NAND(NAND(f,g),1)
1000	f NOR g	(f + g)´	NOR(f,g)
1001	f XNOR g	(f ⊕ g)´	NOR(NOR(f,g´), NOR(f´,g))
1010	NOT f	g´	g´
1011	f ≥ g	f + g´	NOR(NOR(f,g´),1)
1100	NOT g	f´	f´
1101	f ≤ g	f´ + g	NOR(NOR(f´,g),1)
1110	f NAND g	(f . g)´	NOR((f´,g´),1)
1111	1	1	1

I. BDD based design with proper polarity selection for low power

It is seen that in BDD based realization of logic circuits, the area and power consumption is determined by the total number of nodes. If a proper polarity selection of the sub-functions is done than it can not only reduce the number of BDD nodes in the design but also the switching activity of the transistors. For this testing a performance analysis of 4-bit magnitude comparator specially for low power was designed in [5] by developing a general evaluation methodology, and also by BDD computation as well as pre-computation strategy. Fig. 12 shows a 4-bit comparator with second pre-computation architecture.

The experimental results showed that the total product terms was 78 that means 78 node count, but with the help of BDD package tool it reduced to 46 node count. As one node is represented by a 2x1 multiplexer, after synthesizing 2x1 multiplexer in Synopsys tool, the power required for it was 762.3125 nw. As there were a total of 46 nodes so total power taken by 4-bit comparator was found to be 46*762.3125 nw which is equal to 35.0664 µw. With the previous design when the 4-bit magnitude comparator was synthesized in Synopsys tool then the power output was 164.29 µw. Again after applying pre-computation technique in comparator then the total power was 66.6735µw. This was found to be less compared to without applying pre-computation technique in comparator. But still, when all the three ways were compared then one can conclude that, implementation of 4-bit magnitude comparator through BDD is the best way considering the low power aspect.

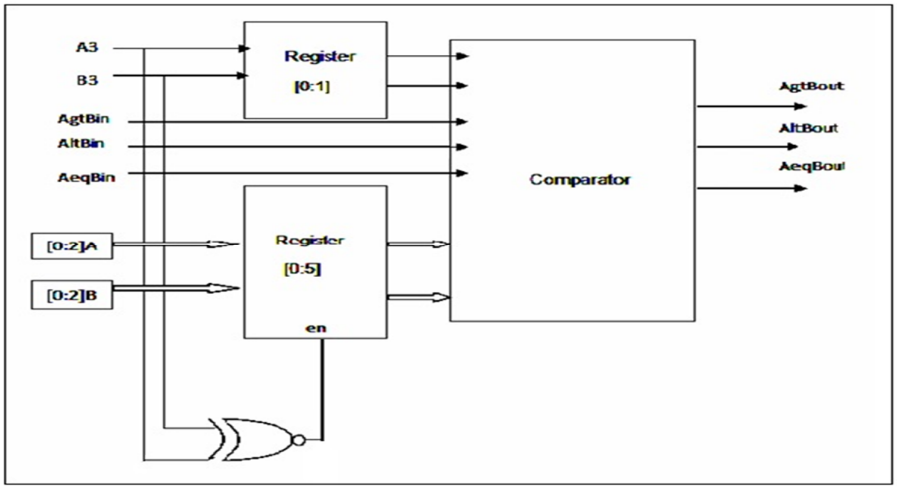

Figure 12. A 4-bit comparator with second pre-computation architecture

III. Conclusion

Through vigorous survey for power reduction using BDDs many different techniques were found out. It is seen that BDD acts as a very efficient tool for power reduction by pre-computing the boolean expression or circuits. After this mux realization is done using different efficient power reduction techniques. This approach is a very good topic for research in low power VLSI design.

References

[1] M. Marik and A. Pal, "Energy-aware Logic Synthesis and Technology Mapping for MUX-based FPGAs", Proceedings of the 17th International Conference on VLSI Design (VLSID'04) IEEE, vol. 4, pp. 1063-9667, 2004.

[2] N. S. Pradhan, G. Paul, A. Pal and B. B. Bhattacharya, "Power Aware BDD-based Logic Synthesis Using Adiabatic Multiplexers", 4th International Conference on Electrical and Computer Engineering ICECE 2006, 19-21 December 2006.

[3] M.A. Gallant, "Synthesis of Low Power CMOS Circuits Using Pass Logic Topology", A thesis presented to the School of Graduate Studies in the Department of Electrical and Cornputer Engineering Royal Military College of Canada Kingston, Ontario, November 1997.

[4] I. Choi , H. Kim, S. Lim, S. Hwang, B. Lee, and B. Kim, "A Kernel-Based Partitioning Algorithm for Low-Power, Low-Area Overhead Circuit Design Using Don't-Care Sets", ETRI Journal, vol. 24, no. 6, December 2002.

[5] P. Singh, C. Gupta and M. Bansal, "Power Optimization In A 4-Bit Magnitude Comparator Circuit Using BDD and Pre-Computation Based Strategy", International Journal of Applied Engineering Research, vol.7, no.11, 2012.

[6] V. Bhagyalakshmi and D.B. Mantri, "A BDD-based Design of An Area-Power Efficient Asynchronous Adder", International Journal of Electronics and Communication Engineering, vol. 6, no.1, pp. 93-103, 2013.

[7] N. S. Pradhan, G. Paul, A. Pal and B. B. Bhattacharya, "Low Power BDD-based Synthesis Using Dual Rail Static DCVSPG Logic", IEEE, vol. 06, pp. 4244-0387, 2006.

[8] D. Somasekhar, Y. Ye, and K.Roy, "An energy recovery static RAM memory core", Proc. IEEE Symp, Low Power Electronics and Design, pp. 62-63, 1995.

[9] V. Bertacco, S. Minato, P. Verplaetse, L. Benini and G. De Micheli, "Decision Diagrams and Pass Transistor Logic Synthesis", Technical Report No.: CSL-TR-97-748, December 1997.

[10] R. Drechsler and M. Thornton, "Fast and Efficient Equivalence Checking based on NAND-BDDs", Proceedings of IFIP International Conference on Very Large Scale Integration, pp. 401-405, 2001.

[11] M. Raseen, A.Assi, P.W. C. Prasad and A. Harb, "BDD Package Based on Boolean NOR Operation", World Academy of Science, Engineering and Technology, vol. 3, 2007.

[12] S. B. Akers, Binary decision diagrams, IEEE Trans. Computers, vol. C-27, no. 6, pp. 509-516, 1978.